AbioCard

I/O Card for BCM2835-based computer

User Manual

August 2012

Table of Contents

1 Features **4**

2 Technical Specifications **4**

3 Installation **5**
AbioCard 5
Enclosure 5
Bill of Materials 6
Assembly Instructions 6

4 Interfacing **7**
Board Overview 7
General-Purpose I/O Connector (K4) 8
Timestamp Input (K6) 8
Analog Inputs Connector (K5) 9
LED Driver Connector (K3) 9
LED Driver Connector (K2) 10

5 I2C Bus **11**
Slave Addresses 11
Real-time Clock 11
8-bit I/O Expander 11
Analog-to-Digital Converter 11
LED Driver 11

6 Software **12**
Program abiocardtime 13
Command Line 13
Source 13
Program abiocardserver 14
Command Line 14
Protocol 14
Source 18
Program abiocardgui 19
User Interface 19
Command Line 22
Module bcm2835_detect 23
API 23
Source 23
Module abiocard 23
Hardware Configuration 23
API 24

Source 26
Distribution Packages 26
Package abiocard 26
Package abiocardgui 27
Installation 28
Package abiocard 28
Package abiocardgui 29
Software License 29
Package abiocard 29
Program abiocardgui 29

7 Application Examples 30
8-bit I/O Expander 30
Basic Input 30
Optically Isolated Input 30
Basic LED Drive Output 30
Output with MOSFET Transistor 31
Output with Bipolar Transistor 31
Optically Isolated Output (low side switch) 31
Optically Isolated Output (high side switch) 32
Outputs with ULN2803 Driver 32
LED driver 33
Basic PWM Output for Driving a LED 33
PWM Output with MOSFET Transistor 33
PWM Output with Bipolar Transistor 33
PWM Outputs with ULN2803 Driver 34

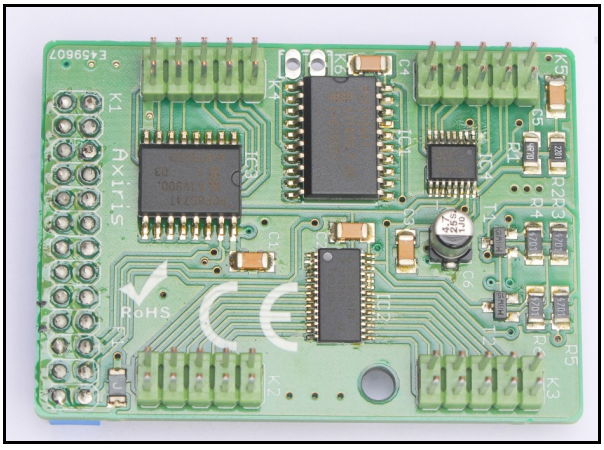
8 Disclaimer 35

9 Contact Information 35

Revision History

Date	Authors	Description
2012-08-20	Peter S'heeren	Initial release.

1 Features



- Battery backed up real-time clock and calendar with integrated temperature compensated crystal oscillator and a 32.768 kHz quartz crystal.
- 8 general-purpose quasi-bidirectional I/O lines, 5 V levels.
- LED driver providing 16 pulse-width modulation (PWM) output channels, 5 V levels @ 97 kHz with LED drive capability (10mA source, 25mA sink).
- Low-power, 12-bit, 8-channel analog-to-digital converter chip featuring internal track/hold, voltage reference, and clock.
- Polyfuse on the 5V line for preventing damage to the computer.
- Compact footprint due to the use of SMD components.
- Free software.

Applications include:

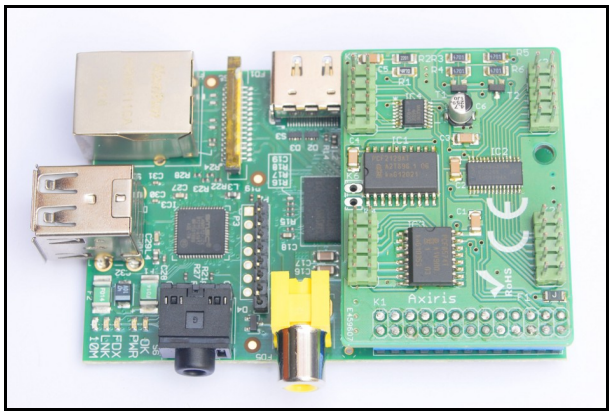
- Home automation (domotics).
- Industrial automation.
- Multimedia (ambient effect, audio spectrum display, ...).
- RGB LED control.
- Educational purposes.

2 Technical Specifications

Dimensions	55 mm x 40 mm x 18 mm (W x D x H)
Weight	16 g (without battery)
RTC backup power	10 years

3 Installation

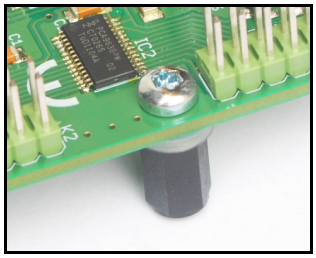
AbioCard

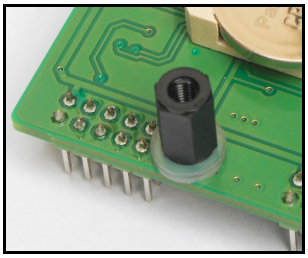


First of all, place a CR2032 battery in the battery holder on the backside of the AbioCard.

Note that when no battery is present, the AbioCard will work except the real-time clock won't keep the time when the system is powered down.

The AbioCard has a 2x13 pin female header that fits perfectly on your BCM2835-based computer. Be sure you connect the AbioCard in such a way it hovers over the computer as shown in the picture to the right.





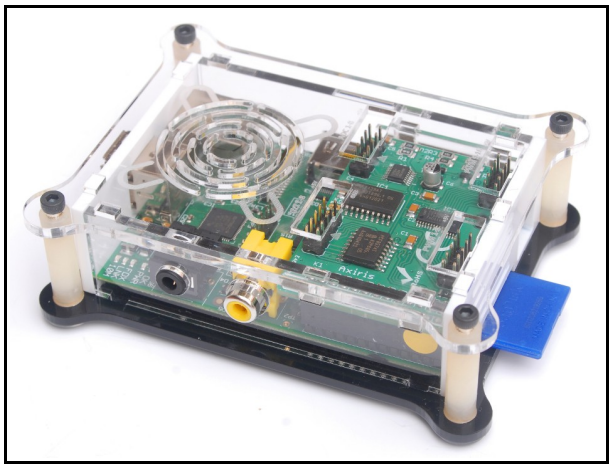
A dedicated enclosure for the combo is available. If you prefer not to use the dedicated enclosure, it's recommended to mount the spacer that came with the AbioCard between the AbioCard and the computer. The AbioCard provides a drill hole for this purpose. Use the included nylon washers to adjust for the optimal height.

Whether you use the enclosure or the spacer, it's important the AbioCard's backside can't touch the computer directly. When mounting, you've to prevent the AbioCard's lithium battery from touching components on the computer PCB.

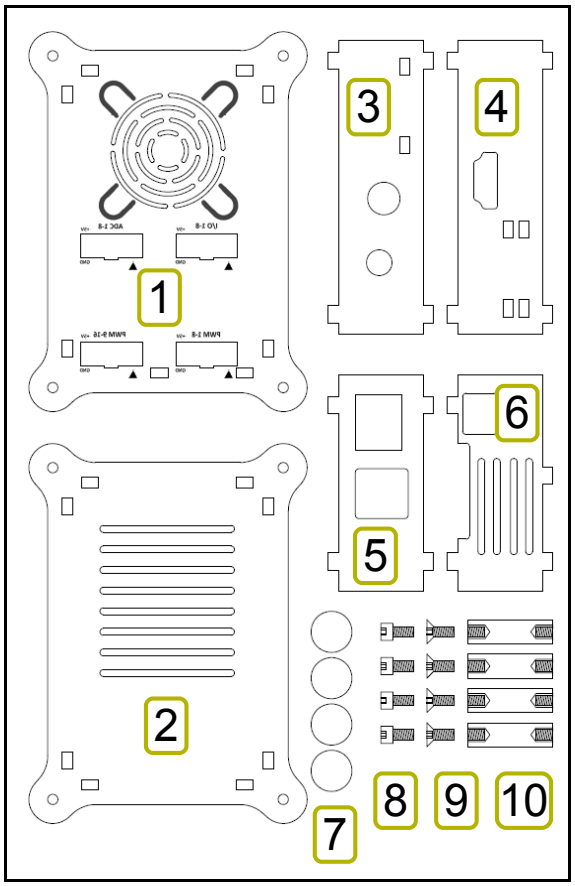
Enclosure

An enclosure specifically designed for the combo is available.

The enclosure comes with holes on top for connecting 2x5 pin female headers. You can use ribbon cable connectors for this purpose.



Bill of Materials



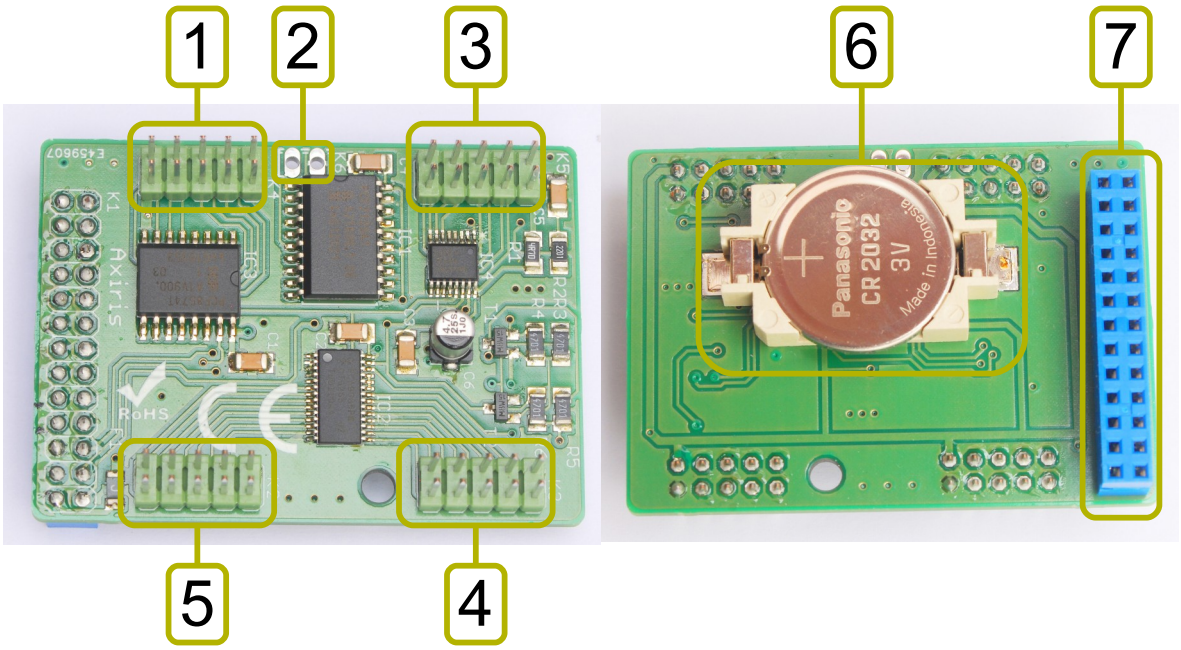
#	Description
1	Transparent acrylic top panel
2	Black acrylic bottom panel
3	Transparent acrylic side panel (audio, video)
4	Transparent acrylic side panel (HDMI)
5	Translucent acrylic side panel (LAN, USB)
6	Translucent acrylic side panel (USB Micro-B)
7	Rubber feet
8	M3 hex. screws top side
9	M3 countersunk hex. screws bottom side
10	Spacers

Assembly Instructions

1. Remove the protective films from the acrylic panels.
2. Mount the AbioCard on the BCM2835-based computer as described earlier.
3. Mount the side panels (3, 4, 5, 6) around the combo.
4. Place the assembly on the bottom panel (2). Make sure the air vents are closest to the LAN and USB panel (5).
5. Mount the top panel (1) on top of the assembly.
6. Insert a spacer (10) in each corner between the top panel and the bottom panel. Drive an M3 hex. screw (8) at the top and an M3 countersunk hex. screw (9) at the bottom. DO NOT exert too much force when tightening the screws!
7. Stick a rubber foot on each M3 countersunk hex. screw (9) on the bottom panel.

4 Interfacing

Board Overview

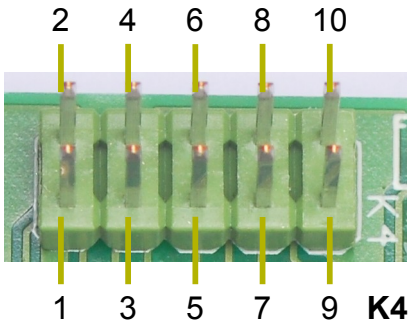


Mark	Label	Description
1	K4	PCF8574 general-purpose I/O connector
2	K6	PCF2129A timestamp input
3	K5	MAX11614 analog inputs connector
4	K3	PCA9635 LED driver pins LED8-LED15 connector
5	K2	PCA9635 LED driver pins LED0-LED7 connector
6		RTC backup power battery, 3 V type CR2032 [1]
7	K1	GPIO female connector

[1] The battery is not included with the product.

General-Purpose I/O Connector (K4)

Mark	Description
1	I/O channel 0
2	I/O channel 1
3	I/O channel 2
4	I/O channel 3
5	I/O channel 4
6	I/O channel 5
7	I/O channel 6
8	I/O channel 7
9	GND
10	5 V output



The I/O channels are quasi-bidirectional. The direction is determined by the software:

- When used as input, software must set the output pin to HIGH.
- When used as output, software writes any value to the pin.

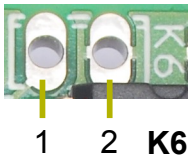
IMPORTANT

The AbioCard doesn't incorporate pull-up resistors on the I/O pins of the PCF8574. If you intend to use an I/O channel as output, and you want to read back the current output logic level, then it is recommended to connect an appropriate pull-up resistor to the pin, else software may read back logic zero rather than the actual state, especially when current is drawn from the pin.

Refer to the PCF8574 datasheet for more information.

Timestamp Input (K6)

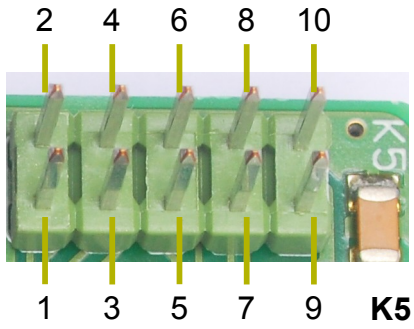
Mark	Description
1	PCF2129A $\overline{TS}$ input
2	GND



Refer to the PCF2129A datasheet for more information about the timestamp functionality.

Analog Inputs Connector (K5)

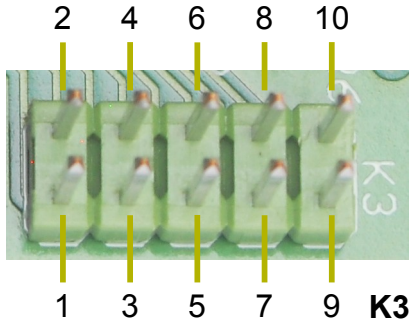
Mark	Description
1	Analog input 0
2	Analog input 1
3	Analog input 2
4	Analog input 3
5	Analog input 4
6	Analog input 5
7	Analog input 6
8	Analog input 7
9	GND
10	5 V output



Connector K5 exposes the 8 analog input pins of the MAX11614. This chip offers a variety of settings for using the analog inputs. Refer to the MAX11614 datasheet for more information.

LED Driver Connector (K3)

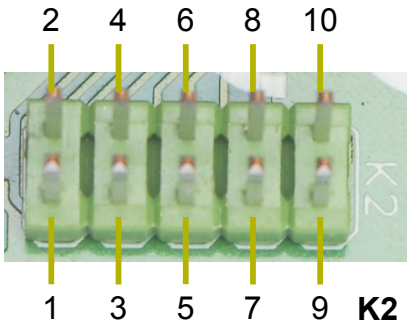
Mark	Description
1	LED driver pin LED8
2	LED driver pin LED9
3	LED driver pin LED10
4	LED driver pin LED11
5	LED driver pin LED12
6	LED driver pin LED13
7	LED driver pin LED14
8	LED driver pin LED15
9	GND
10	5 V output



Connector K3 exposes the last eight LED outputs of the LED driver.

LED Driver Connector (K2)

Mark	Description
1	LED driver pin LED0
2	LED driver pin LED1
3	LED driver pin LED2
4	LED driver pin LED3
5	LED driver pin LED4
6	LED driver pin LED5
7	LED driver pin LED6
8	LED driver pin LED7
9	GND
10	5 V output



Connector K2 exposes the first eight LED outputs of the LED driver.

5 I2C Bus

Slave Addresses

The I2C bus on the AbioCard interconnects your BCM2835-based computer with a set of I2C slave devices. Each slave device responds to one or more I2C addresses. The slave addresses are:

Address	Slave Device
1010001b	PCF2129A real-time clock
0100111b	PCF8574 8-bit I/O expander
0110011b	MAX11614 analog-to-digital converter
0001000b	PCA9635 LED driver
1110000b	PCA9635 all devices "LED All Call" address
0000011b	PCA9635 all devices SWRST (software reset)

Real-time Clock

The PCF2129A is a real-time clock and calendar with an integrated temperature compensated crystal oscillator and a 32.768 kHz quartz crystal. The PCF2129A has a backup battery switch-over circuit, a programmable watchdog function, a timestamp function, and many other features.

The board has a holder for a CR2032 battery that functions as backup power for the PCF2129A.

8-bit I/O Expander

The PCF8574 is an 8-bit I/O expander. The outputs feature a high current drive capable of directly driving LEDs.

The I/O channels are quasi-bidirectional. Refer to the PCF8574 datasheet for more information.

Analog-to-Digital Converter

The Maxim MAX11614EEE+ is a low-power, 12-bit, 8-channel analog-to-digital converter chip featuring internal track/hold (T/H), voltage reference, and clock.

LED Driver

The PCA9635 is a LED driver optimized for red/green/blue/amber (RGBA) color mixing applications.

6 Software

The AbioCard comes with a number of software programs that run on your BCM2835-based computer with Linux for ARM installed. The **abiocardgui** program runs on other computer systems and operating systems.

IMPORTANT!

The programs that run on your BCM2835-based computer directly communicate with the AbioCard from user-mode using the first serial controller (BSC0) on the BCM2835 application processor. Therefore it is important that no I2C driver is loaded in the Linux operating system.

This document makes the following assumptions:

- You've installed Debian6 Linux or a derivative distribution on the SD card of your BCM2835-based computer.
- You've a personal log in that's not root.
- You know how to run programs as root (**sudo** command, setuid access right flag).
- You know how to log in as root (log in prompt, the **su** command).

This document describes commands entered at the command prompt. When entering a command at the prompt of your personal log in, the document uses the $ notation. For example:

```
$ sudo ./abiocardtime
```

A command entered at the prompt of the root log in is written with the # notation. For example:

```
# ./abiocardtime
```

Program abiocardtime

This program runs exclusively on a BCM2835-based computer and needs root permissions. It only runs for a very short time. It reads the current date and time from the RTC on the AbioCard and updates the operating system's software clock accordingly.

The program can't set the operating system's software clock when the real-time clock reports the power-up detected state. You'll need to set the actual date and time in the real-time clock first. You can use the **abiocardgui** program for this purpose or run the **abiocardtime** program at the command prompt with the "-s" parameter.

The program is expected to run once during Linux startup. Once Linux is running, it's recommended to execute the program periodically in order to keep the operating system's software clock in line with the more accurate real-time clock on the AbioCard.

Command Line

Parameter	Description
-s STRING	Set date and time in the RTC. Example STRING: "2012-08-18 20:24:48".
-u	Update the system time using the RTC.
-p	Poll the RTC and print.

When multiple commands are specified, they're executed in the following order:

```
-s STRING ► -u ► -p
```

Example execution of the program as root. Command entered in the program's own directory:

```
# ./abiocardtime -u
```

Source

Filename	Description
`bcm2835_detect.c`	Module **bcm2835_detect**.
`bcm2835_detect.h`	
`abiocard.c`	Module **abiocard**.
`abiocard.h`	
`abiocard_data.h`	
`abiocardtime.c`	The **abiocardtime** program.
`platform.h`	Platform-specific declarations.
`abiocardtime.mk`	Makefile for the **abiocardtime** program.

To build the program:

```
$ make -f abiocardtime.mk
```

To clean up the intermediate build files:

```
$ make -f abiocardtime.mk clean
```

Program abiocardserver

This server program runs exclusively on a BCM2835-based computer and needs root permissions. It offers a socket interface for communicating with the AbioCard over the network. The server supports the hardware configuration implemented by the **abiocard** module.

The dearest client of the server is the **abiocardgui** program.

The server uses a simple protocol for accepting commands and generating responses in human-readable format using the ASCII character set. There's a timeout on incoming characters, specified in seconds. When the server doesn't receive any character for the specified time, it'll close the connection with the client.

You can use a terminal program (like Putty) for sending commands manually to the server. Be sure to specify a timeout value big enough so the server won't shut down your connection should you type too slow. You may as well specify the maximum timeout value for the purpose of using a terminal program.

Since only one AbioCard can be plugged into your BCM2835-based computer, the server will accept one incoming socket connection only.

The server is designed to run in the background. When you require the permanent services of the program, it's recommended to execute **abiocardserver** during Linux startup.

The server will only interact with the AbioCard when it has accepted an incoming connection. As such it can safely be invoked alongside the **abiocardtime** program during Linux startup without the risk of interfering with the proper execution of the **abiocardtime** program.

Command Line

Parameter	Description
-p n	Specifies the port number the server must listen to. Value n=1..65535 decimal. This parameter is mandatory.
-t n	Time-out value between two incoming characters. Value n=5..65535 decimal (seconds). This parameter is optional. The default value is 30 seconds.

Example execution of the server. The server listens to port 5678. Time-out value is set to 5 seconds. Command entered in the program's own directory:

```
$ sudo ./abiocardserver -p 5678 -t 5
```

Protocol

The server accepts commands and returns a response if defined. The server never sends

a response autonomously, it only responds to individual commands.

The server will only respond to a command when the command was correctly formatted, was executed successfully, and a response is defined.

Both commands and responses are composed of a string of ASCII characters concluded by an end-of-line character. The server recognizes LF (10) and CR (13) as end-of-line marker. A response from the server is always concluded with a LF (10).

The following formatting is used in the description of the commands and responses:

Format	Description
CR	Literal characters i.e. character 'C' followed by character 'R'.
ABC	These characters containing specific information as explained.
[ABC]	An array of these characters containing specific information as explained.
<EOL>	End-of-line marker, either LF (10) or CR (13).
<LF>	End-of-line marker LF (10).
<none>	No response.

CR – Clock Read

Command	**CR<EOL>**	
Response	**CR**YYMMDDhhmmsspb**<LF>**	
	YY	The year, 00..99 decimal, meaning year 2000..2099.
	MM	The month, 01..12 decimal, meaning January..December.
	DD	The day, 01..31 decimal.
	hh	The hour, 00..23 decimal.
	mm	The minute, 00..59 decimal.
	ss	The second, 00..59 decimal.
	p	The power-up detected flag: ▪ =0: No power-up detected, date and time fields are valid. ▪ =1: Power-up detected, **date and time fields are invalid**.
	b	The battery low flag, either 0 or 1.

This command reads (“polls”) the real-time clock.

The date and time field are valid only when no power-up is detected. To clear the power-up detected state, data and time must be set by sending a CW command.

The battery is deemed low when either the battery is running out or when there's no battery at all.

CW – Clock Write

Command	**CW**YYMMDDhhmmss**<EOL>**	
	YY	The year, 00..99 decimal, meaning year 2000..2099.
	MM	The month, 01..12 decimal, meaning January..December.
	DD	The day, 01..31 decimal.
	hh	The hour, 00..23 decimal.
	mm	The minute, 00..59 decimal.
	ss	The second, 00..59 decimal.
Response	**<none>**	

This command writes the date and time to the real-time clock.

If successful, the power-up detected state will be cleared.

AR – ADC Read

Command	**AR<EOL>**	
Response	**AR**V00V01V02V03V04V05V06V07**<LF>**	
	V00	Converted value of analog input 0, 000..FFF hexadecimal.
	V01	Converted value of analog input 1, 000..FFF hexadecimal.
	V02	Converted value of analog input 2, 000..FFF hexadecimal.
	V03	Converted value of analog input 3, 000..FFF hexadecimal.
	V04	Converted value of analog input 4, 000..FFF hexadecimal.
	V05	Converted value of analog input 5, 000..FFF hexadecimal.
	V06	Converted value of analog input 6, 000..FFF hexadecimal.
	V07	Converted value of analog input 7, 000..FFF hexadecimal.

This command converts all eight analog inputs and responds with the values.

ER – I/O Expander Read

Command	**ER<EOL>**	
Response	**ER**XX**<LF>**	
	XX	8-bit value read from the I/O expander inputs, 00..FF hexadecimal.

This command reads the state of the inputs of the I/O expander chip and responds with the resulting value.

EW – I/O Expander Write

Command	**EW**XX**<EOL>**	
	XX	8-bit value to write to the I/O expander outputs, 00..FF hexadecimal.
Response	**<none>**	

This command writes the 8-bit value to the outputs of the I/O expander.

PR – PWM Read

Command	**PR**SSCC**<EOL>**	
	SS	Start index, 00..10 hexadecimal.
	CC	Count, 01..11 hexadecimal.
Response	**PR**SSCC**[**XX**]<LF>**	
	SS	Start index, 00..10 hexadecimal.
	CC	Count, 01..11 hexadecimal.
	[XX]	Array of register values, 00..FF hexadecimal. The count field indicates the number of array elements.

This command reads registers in the LED driver.

Indexes 0 to 15 correspond with registers PWM0 to PWM15, index 16 with register GRPPWM in the LED driver chip.

PW – PWM Write

Command	**PW**SSCC**[**XX**]<EOL>**	
	SS	Start index, 00..10 hexadecimal.
	CC	Count, 01..11 hexadecimal.
	[XX]	Array of register values, 00..FF hexadecimal. The count field indicates the number of array elements.
Response	**<none>**	

This command writes registers in the LED driver.

Indexes 0 to 15 correspond with registers PWM0 to PWM15, index 16 with register GRPPWM in the LED driver chip.

HI – Hardware Information

Command	**HI<EOL>**	
Response	**HI**XX**<LF>**	
	XX	Hardware presence bit mask, 00..FF hexadecimal. Bit 0: Real-time clock is present yes/no (1/0). Bit 1: I/O expander is present yes/no (1/0). Bit 2: Analog-to-digital converter is present yes/no (1/0). Bit 3: LED driver is present yes/no (1/0). Bit 4..7: Reserved (0).

This command queries the hardware information gathered during initialization of the AbioCard.

QU – Quit the server

Command	**QU<EOL>**
Response	**<none>**

When the server receives this command, it immediately closes the connection, shuts down the port it's listening to, and terminates.

Source

Filename	Description
`bcm2835_detect.c`	Module **bcm2835_detect**.
`bcm2835_detect.h`	
`abiocard.c`	Module **abiocard**.
`abiocard.h`	
`abiocard_data.h`	
`abiocardserver.c`	The **abiocardserver** program.
`platform.h`	Platform-specific declarations.
`abiocardserver.mk`	Makefile for the **abiocardserver** program.

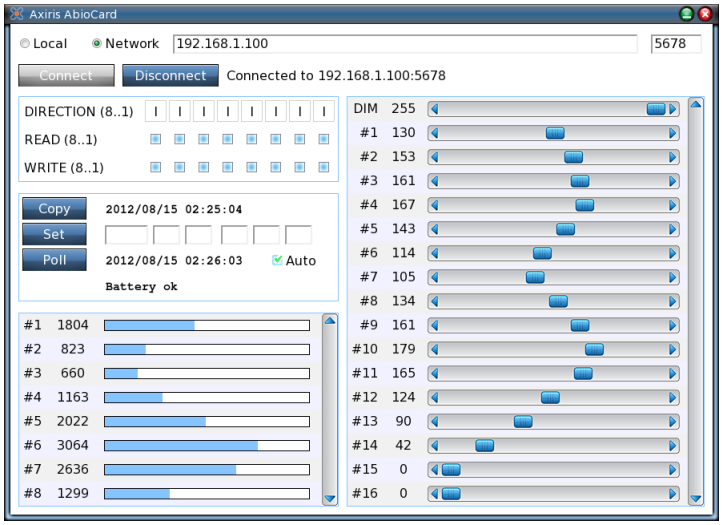
To build the program:

```
$ make -f abiocardserver.mk
```

To clean up the intermediate build files:

```
$ make -f abiocardserver.mk clean
```

Program abiocardgui



The **abiocardgui** program is a graphical front-end application that visualizes the state of the AbioCard and allows the user to interact with the AbioCard.

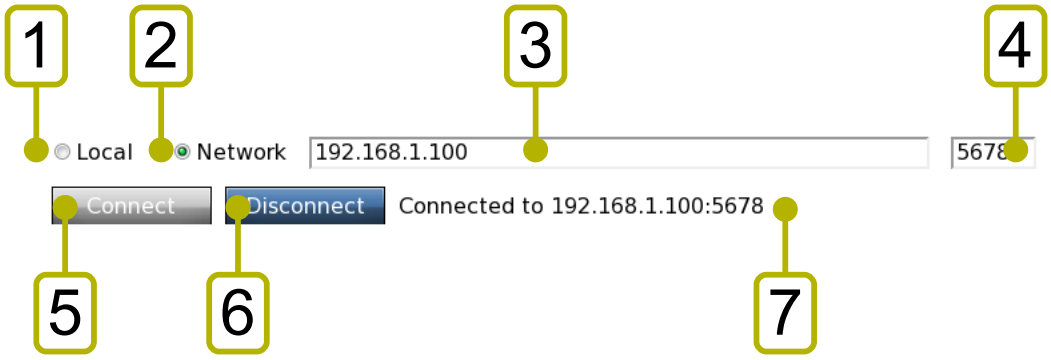
The program runs on a variety of systems including Linux for x86 and ARM, and Windows 2000 and later.

When the program runs locally on your BCM2835-based computer, it can directly communicate with the AbioCard.

The program is capable of connecting to the **abiocardserver** program over a network. This mode allows you to inspect and control the AbioCard remotely, thus from another computer.

User Interface

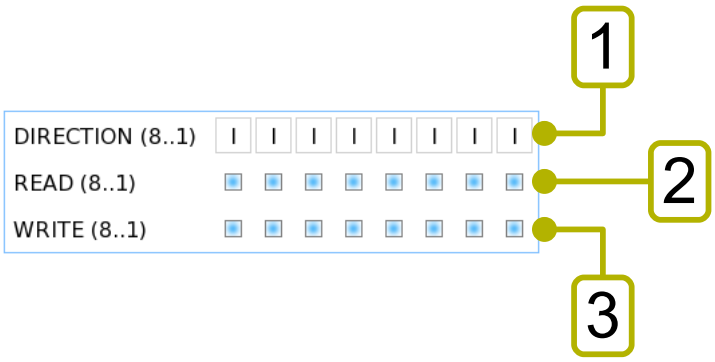
The Interface Panel



Mark	Description
1	Click to select local I/O with the AbioCard.
2	Click to select network I/O with the AbioCard.
3	For network I/O, you've to enter the host here. The application accepts three representations: ▪ An IPv4 address. ▪ The word **localhost**. It's interpreted as IPv4 address 127.0.0.1. ▪ A domain name. Upon connection, the application will perform a DNS look up to determine the corresponding IPv4 address.

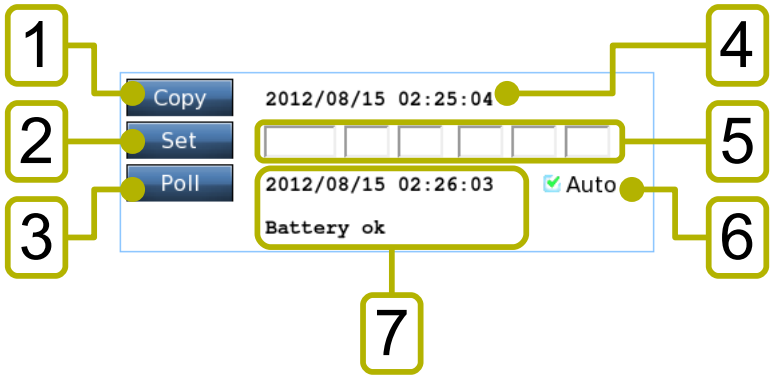
4	For network I/O, you've to enter the port number here.
5	Click the Connect button to connect with the AbioCard.
6	Click the Disconnect button to disconnect from the AbioCard.
7	Status information is displayed here.

The I/O Expander Panel



Mark	Description
1	Click the buttons to toggle the direction of the I/O channel.
2	State of the I/O lines.
3	For I/O channels with output direction, click to toggle the output state.

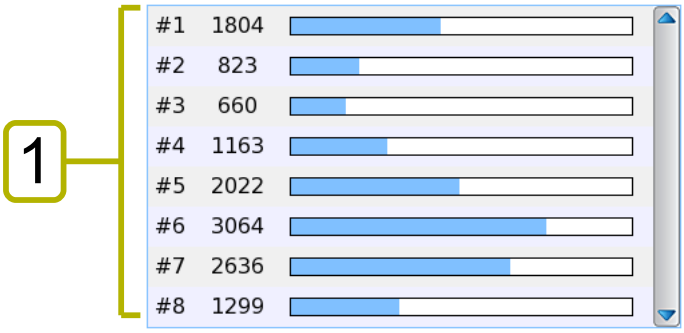
The Real-time Clock Panel



Mark	Description
1	Click the Copy button to copy the system time to the edit fields for data and time. The system time is displayed next to the Copy button.
2	Click the Set button to write the date and time in the edit fields to the AbioCard.

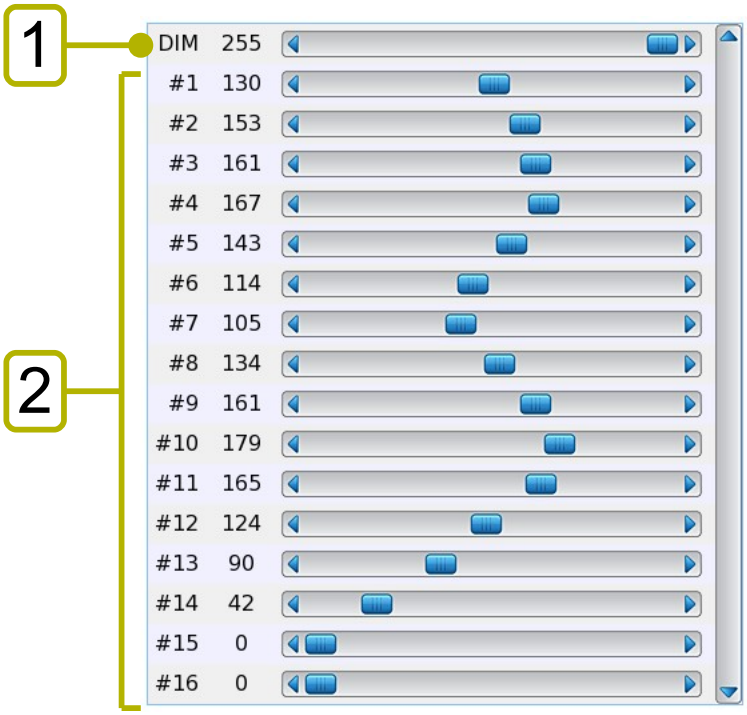
3	Click the Poll button to force a clock read command with the AbioCard. This button is useful when the Auto-poll checkbox is unchecked.
4	The current system time.
5	These edit fields enable you to enter a date and time which you can set in the AbioCard device using the Set button.
6	The Auto-poll checkbox let you decide whether the application automatically polls the real-time clock.
7	The information most recently read from the real-time clock.

The Analog-to-Digital Converter Panel



Mark	Description
1	The state of the analog inputs.

The LED Driver Panel



Mark	Description
1	The state of the PWMGRP register. Slide the scrollbar to adjust the value.
2	The state of registers PWM0 to PWM15. Slide a scrollbar to adjust the value.

Command Line

Not applicable. The program ignores the command line.

Module bcm2835_detect

The computer is built around the BCM2835 application processor. The BCM2835 is a member of the BCM2708 family of microchips. The **bcm2835_detect** module is designed to detect a BCM2835-based computer.

All programs that communicate with the AbioCard access the BCM2835's hardware registers directly from user-mode using memory-mapped I/O. Running these programs on a system other than a BCM2835-based computer will lead to unpredictable results unless they're aware of the presence or non-presence of the expected hardware. To guarantee such problems won't occur, the programs incorporate the **bcm2835_detect** module. They will first detect the BCM2835 hardware before they make any attempt to access the AbioCard.

API

Each function in the API is briefly explained. Refer to the source code for more details.

```
FLAG  bcm2835_detect (FLAG *res_detected)
```

Runs the detection procedure.

Returns one if the detection procedure was executed successfully, zero elsewise. If successful, the `res_detected` field reflects the actual result of the detection. Field `res_detected` will be set to one if a BCM2835-based computer is detected, zero if not.

Source

Filename	Description
`bcm2835_detect.c`	Implementation of the module.
`bcm2835_detect.h`	Public header of the module.
`platform.h`	Platform-specific declarations.

Module abiocard

The **abiocard** module offers an API in C for communicating with the AbioCard from your own program.

The **abiocard** module needs to run with root permissions, thus in the context of the root. If not, the initialisation function will return an error code.

Hardware Configuration

The chips on the AbioCard offer many features and configurations. To keep the software simple, the **abiocard** module offers a broad subset of the AbioCard's capabilities through its API. The programs **abiocardserver** and **abiocardgui** support the same subset.

Chip	Supported subset
Real-time clock	▪ Power-up detection during polling. ▪ Low battery detection during polling. ▪ Current time and date function, 24-hour mode only, weekday not used.
8-bit I/O expander	▪ Not configurable.

LED driver	▪ All LED outputs programmable.
	▪ All LED groups under dimming control, no blinking.
	▪ Software handling of dimming value:
	  ▪ When GRPPWM is 0..254, all LED group are set to mode 3.
	  ▪ When GRPPWM is 255, all LED group are set to mode 2.
	▪ INVRT=1: Output logic state inverted.
	▪ OUTDRV=1: Totem-pole structure.
	▪ DMBLNK=0: Group control is dimming.
	▪ OCH=0: Outputs change on I2C STOP command.
	▪ OUTNE[1..0]=10b: When $\overline{OE}$ is high, LED outputs are high-impedance. Note that $\overline{OE}$ is hardwired to ground.

Analog-to-digital converter	▪ Unipolar: sampling values are 12-bit unsigned (0..4095). ▪ Single-ended use of the analog inputs. ▪ All 8 analog inputs are converted in one command. ▪ Using internal clock. ▪ Using internal voltage reference (4.096V).

API

The API support the hardware configuration described in the previous chapter.

Each function in the API is briefly explained. Refer to the source code for more details.

```
FLAG  abiocard_init (ABIOCARD_INIT_IO *io)
```

Initializes the module. This function performs the following tasks:

- It tries to acquire the lock file, so only one software instance as access to the AbioCard.
- It maps the required registers of the BCM2835 in memory and initializes the corresponding hardware components on the application processor.
- It probes for the presence of the various chips on the AbioCard. At least one chip must be present, else initialization fails.
- It initializes the LED driver chip and analog-to-digital converter chip.

When the function returns one, the initialization was successful and the other functions can be called safely.

When the function returns zero, there was an error. In this case all other functions except for **`abiocard_deinit`** mustn't be called. The **`io->err`** field will contain the error code.

```
VOID  abiocard_deinit (VOID)
```

Deinitializes the module. A program must call this function when it's done using the

module.

This functions may be called multiple times. This function may be called before and after invoking **abiocard_init**.

```
FLAG  abiocard_ioexp_write (U8 word)
```

Writes the given 8-bit value to the I/O pins of the I/O expander chip.

Returns one when successful, zero when something went wrong.

```
FLAG  abiocard_ioexp_read (U8 *word)
```

Reads the I/O pins of the I/O expander chip and writes the resulting 8-bit value at the address pointed to by **word**.

Returns one when successful, zero when something went wrong.

```
FLAG  abiocard_rtc_poll (ABIOCARD_RTC_POLL_INFO *info)
```

Polls the real-time clock. The results are written to the buffer pointed to by **info**.

Returns one when successful, zero when something went wrong.

```
FLAG  abiocard_rtc_set_time (ABIOCARD_RTC_TIME *time)
```

Sets the given date and time in the real-time clock.

Returns one when successful, zero when something went wrong.

```
FLAG  abiocard_pwm_write_range (U8 *buf, U8 start, U8 cnt)
```

Writes a range of values to the LED driver chip. The values are written to indexes numbered 0 to 16. Parameter **start** is the first index, parameter **cnt** the number of values to write. Parameter **buf** points to the source buffer.

Indexes 0 to 15 correspond with registers PWM0 to PWM15, index 16 with register GRPPWM in the LED driver chip.

Returns one when successful, zero when something went wrong.

```
FLAG  abiocard_pwm_read_range (U8 *buf, U8 start, U8 cnt)
```

Reads a range of values from the LED driver chip. The values are read from indexes numbered 0 to 16. See function **abiocard_pwm_write_range** for a description of the meaning of the indexes.

Returns one when successful, zero when something went wrong.

```
FLAG  abiocard_adc_convert (ABIOCARD_ADC_CNV_DATA *buf)
```

Converts all analog inputs and stores the resulting values in the buffer pointed to by **buf**.

Returns one when successful, zero when something went wrong.

Source

Filename	Description
abiocard.c	Implementation of the module.
abiocard.h	Public header of the module. Includes **abiocard_data.h**.
abiocard_data.h	Public header with only data declarations.
platform.h	Platform-specific declarations.

Distribution Packages

Package abiocard

The **abiocard** package contains the software that is essential for working with the AbioCard.

The package contains two executables: **abiocardserver** and **abiocardtime**. Their build target is armv6l armel, they run on armv6l armhf as well. These executable files are included for in case your system doesn't have the GCC tool chain.

abiocard.tar.gz	
▷ abiocard	
▶ abiocard.c	Module **abiocard**.
▶ abiocard.h	
▶ abiocard_data.h	
▶ abiocard_test_1.c	AbioCard test 1. An example program how-to use the **abiocard** module.
▶ abiocard_test_1.mk	
▶ abiocardserver	AbioCard server (binary, armv6l, armel).
▶ abiocardserver.c	
▶ abiocardserver.mk	
▶ abiocardtime	AbioCard time (binary, armv6l, armel).
▶ abiocardtime.c	
▶ abiocardtime.mk	
▶ bcm2835_detect.c	Module **bcm2835_detect**.
▶ bcm2835_detect.h	
▶ install.sh	Installation script. You've to run the **install.sh** script in the context of your user account, not the root account (except when you're installing as the root user).
▶ install_root.sh	
▶ platform.h	Platform-specific declarations.

Package abiocardgui

The **abiocardgui** package is supplied for various operating systems.

abiocardgui-linux-armv6l

abiocardgui-linux-armv6l.tar.gz	
▷ abiocardgui-linux-armv6l	
▶ abiocardgui	AbioCard GUI (binary, armv6l, armel).
▶ abiocardgui.cfg	Configuration file.
▶ abiocardgui.desktop	Desktop file, compliant with the Desktop Entry Specification by the **freedesktop.org** organization.
▶ abiocardgui.ico	Icon file displayed on your desktop.
▶ DejaVuSans.ttf	Font file.
▶ FreeMonoBold.otf	Font file.
▶ install.sh	Installation script. You've to run the **install.sh** script in the context of your user account, not the root account (except when you're installing as the root user).
▶ install_root.sh	
▶ libfreetype.so.6	FreeType library.

abiocardgui-linux-i686

abiocardgui-linux-i686.tar.gz	
▷ abiocardgui-linux-i686	
▶ abiocardgui	AbioCard GUI (binary, i686).
▶ abiocardgui.cfg	Configuration file.
▶ abiocardgui.desktop	Desktop file, compliant with the Desktop Entry Specification by the **freedesktop.org** organization.
▶ abiocardgui.ico	Icon file displayed on your desktop.
▶ DejaVuSans.ttf	Font file.
▶ FreeMonoBold.otf	Font file.
▶ install.sh	Installation script. You've to run the **install.sh** script in the context of your user account, not the root account (except when you're installing as the root user).
▶ install_root.sh	
▶ libfreetype.so.6	FreeType library.

abiocardgui-linux-x86_64

abiocardgui-linux-x86_64.tar.gz	
▷ abiocardgui-linux-x86_64	
▶ abiocardgui	AbioCard GUI (binary, x86_64).
▶ abiocardgui.cfg	Configuration file.
▶ abiocardgui.desktop	Desktop file, compliant with the Desktop Entry Specification by the **freedesktop.org** organization.
▶ abiocardgui.ico	Icon file displayed on your desktop.
▶ DejaVuSans.ttf	Font file.
▶ FreeMonoBold.otf	Font file.
▶ install.sh	Installation script. You've to run the **install.sh** script in the context of your user account, not the root account (except when you're installing as the root user).
▶ install_root.sh	
▶ libfreetype.so.6	FreeType library.

abiocardgui-win32

abiocardgui-win32.zip	
▷ abiocardgui-win32	
▶ abiocardgui.exe	AbioCard GUI (binary, Win32).
▶ abiocardgui.cfg	Configuration file.
▶ DejaVuSans.ttf	Font file.
▶ FreeMonoBold.otf	Font file.
▶ freetype6.dll	FreeType library.
▶ zlib1.dll	Zip library.

Installation

Package abiocard

Unpack the **abiocard** package in a local directory in your home directory on your BCM2835-based computer.

Run the **install.sh** script in the context of your user account to install the **abiocard** group of programs in Linux.

Be sure to run the installation script from the command line rather than the file manager. The script will **sudo** and as such the system may ask for the root password.

The installation script will add three lines to the cron table of the root user:

- Run the **abiocardtime** program at reboot.
- Run the **abiocardtime** program daily.
- Start the **abiocardserver** program at reboot (port 5678, timeout 20 seconds).

The installation script will install the programs in directory **/opt/abiocard**.

To set the date and time of the real-time clock and your operating system, run the following command with the correct date and time set with the **-s** parameter:

```
$ /opt/abiocard/abiocardtime -u -s "2012-08-18 20:24:48"
```

Package abiocardgui

Linux

Unpack the **abiocardgui** package in a local directory in your home directory. Be sure you pick the distribution that corresponds with your operating system.

Run the **install.sh** script in the context of your user account to install the **abiocardgui** program in Linux.

Be sure to run the installation script from the command line rather than the file manager. The script will **sudo** and as such the system may ask for the root password.

The installation script will install the files in directory **/opt/abiocardgui**.

The installation script will add a icon on your desktop by placing a desktop file in the Desktop directory in your home directory. This feature works only if your window manager is compliant with the Desktop Entry Specification by the **freedesktop.org** organization.

Windows

Unpack the **abiocardgui** package for Win32 in a local directory on your system.

Execute **abiocardgui.exe** to run the application. You may create a shortcut to the executable file on your desktop.

Software License

Package abiocard

The license is stated in the source text files.

Program abiocardgui

The computer program **abiocardgui** is free of charge ("freeware").

It's allowed to copy or distribute the software providing that you do so with the distribution in its entirety as described in chapter Distribution. It's forbidden to change the software or any other files that are part of the distribution.

It's forbidden to sell, rent, or profit from the distribution in its entirety or the software or any other files that are part of the distribution.

It's allowed to spread the distribution in a packaging format different from the packaging format provided by Axiris.

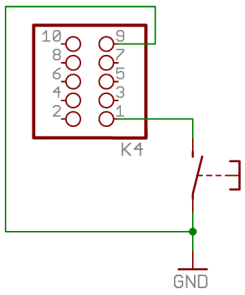
7 Application Examples

8-bit I/O Expander

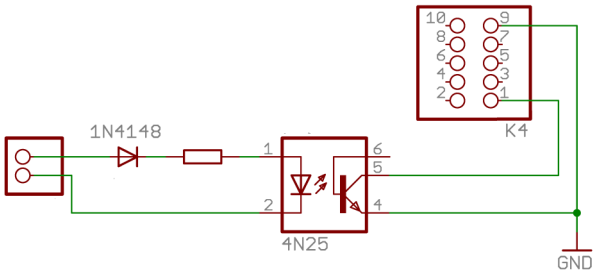
Basic Input

This is a minimalistic circuit for using an input. When the button is unpressed, the input is internally pulled to 5 V and is read as HIGH. When the button is pressed, the input is pulled to ground and is read as LOW.

Software must set the I/O pin to HIGH to use it as an input.



Optically Isolated Input

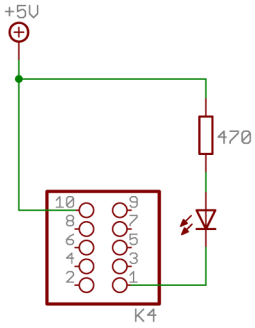


In this example, the input is optically isolated from the control circuit.

Software must set the I/O pin to HIGH to use it as an input.

Basic LED Drive Output

This example circuit shows an output driving a LED. When the output is LOW, the LED will light up. When the output is HIGH, the LED is off.



IMPORTANT!

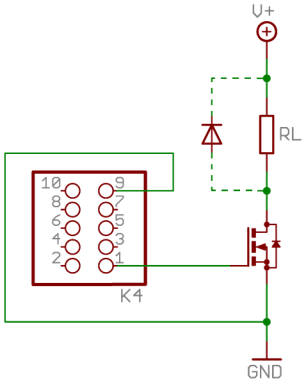
- You must choose the resistor value depending on the type of LED and the expected light intensity.
- The LED mustn't draw more than 25 mA. If multiple outputs are connected to LEDs as shown in the picture, than the total current mustn't exceed 100 mA. See the datasheet for more information.

Output with MOSFET Transistor

This example circuit shows how to switch a load using an enhancement-mode N-channel MOSFET.

It's important to use a logic-level MOSFET that switches on reliably at 5 V gate voltage. Example MOSFETs include 2N7000, IRL520, IRFD220, BTS141.

The maximum load current and load voltage depend on the type of MOSFET. For inductive loads, it's recommended to add a clamp diode as depicted.

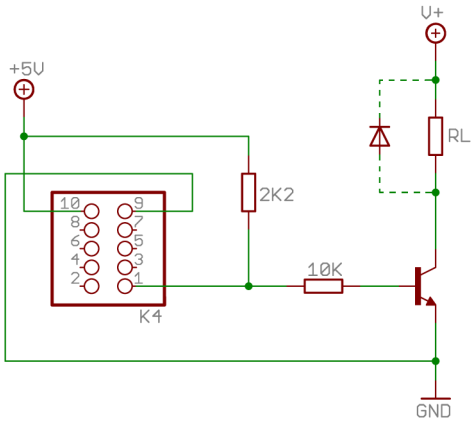


Output with Bipolar Transistor

This example circuit shows how to switch a load using an NPN bipolar transistor. You can replace the NPN transistor with a NPN Darlington transistor to increase the current amplification.

The maximum load current and load voltage depend on the type of transistor. For inductive loads, it's recommended to add a clamp diode as depicted.

The maximum current sourced by the I/O expander chip is 100 µA. It's recommended to add an additional pull-up resistor to increase the base current through the transistor. The pull-up resister also guarantees the I/O expander correctly reads back a HIGH level when software sets the output HIGH.

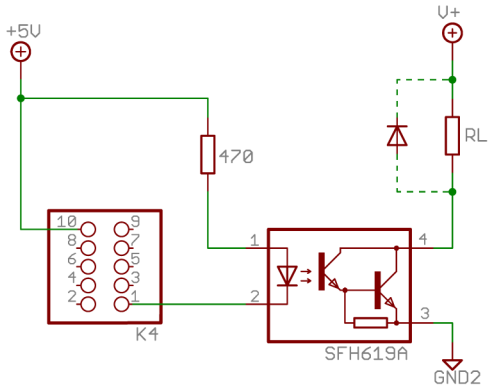


The set-up with the MOSFET is recommended over this example circuit.

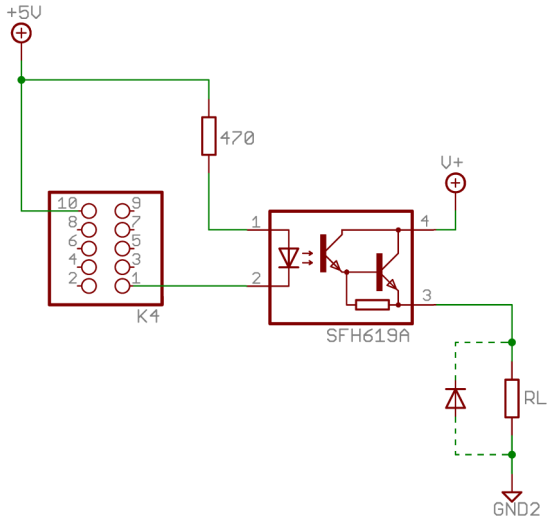
Optically Isolated Output (low side switch)

This example shows an optically isolated output (low side switch).

The optocoupler used in this example is able to switch 125 mA maximum at 300 V maximum.

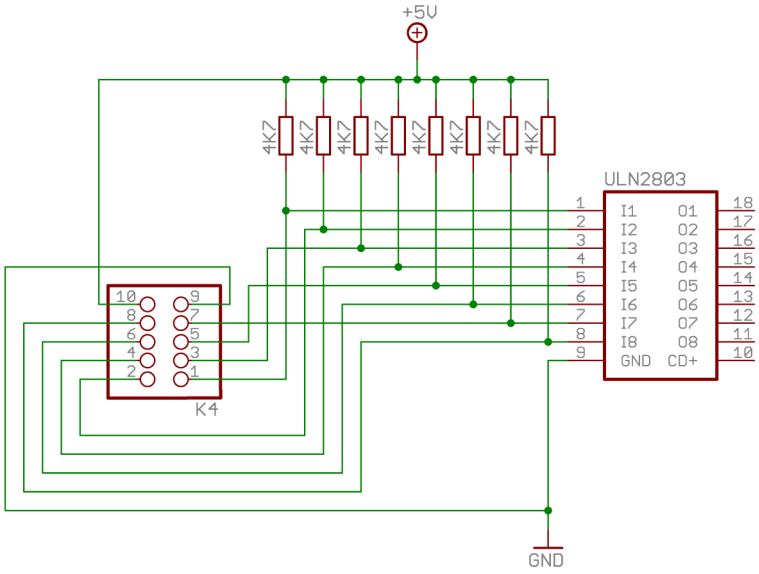


Optically Isolated Output (high side switch)



This example shows an optically isolated output (high side switch).

Outputs with ULN2803 Driver



In this example, all eight output are connected to a ULN2803 driver.

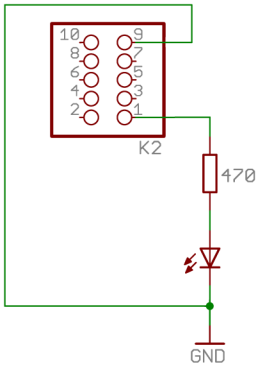
The I/O expander chip can only source 100 µA per output. The pull-up resistors are required to reliably drive the inputs on the ULN2803.

LED driver

Basic PWM Output for Driving a LED

This example circuit shows how to control the intensity of a LED using an output of the LED driver.

The maximum LED current mustn't exceed 10 mA.

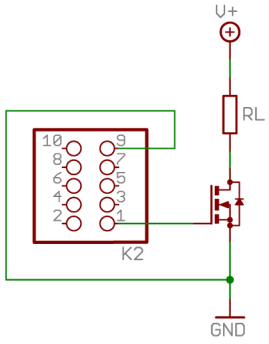


PWM Output with MOSFET Transistor

This example circuit shows how to switch a load using an enhancement-mode N-channel MOSFET.

It's important to use a logic-level MOSFET that switches on reliably at 5 V gate voltage. Example MOSFETs include 2N7000, IRL520, IRFD220, BTS141.

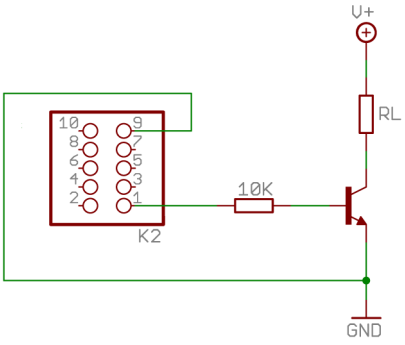
The maximum load current and load voltage depend on the type of MOSFET. It's not recommended to connect inductive loads.



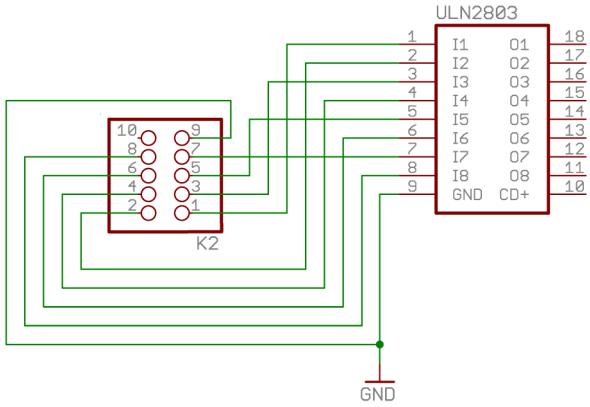
PWM Output with Bipolar Transistor

This example circuit shows how to control the intensity of a LED using an NPN bipolar transistor.

The maximum load current depends on the type of transistor.



PWM Outputs with ULN2803 Driver



In this example, all eight outputs are connected to a ULN2803 driver.

Since the LED driver chip can source enough current to drive the ULN2803, pull-up resistors are not required.

8 Disclaimer

Axiris products are not designed, authorized or warranted to be suitable for use in space, nautical, space, military, medical, life-critical or safety-critical devices or equipment.

Axiris products are not designed, authorized or warranted to be suitable for use in applications where failure or malfunction of an Axiris product can result in personal injury, death, property damage or environmental damage.

Axiris accepts no liability for inclusion or use of Axiris products in such applications and such inclusion or use is at the customer's own risk. Should the customer use Axiris products for such application, the customer shall indemnify and hold Axiris harmless against all claims and damages.

9 Contact Information

Official website: http://www.axiris.be/